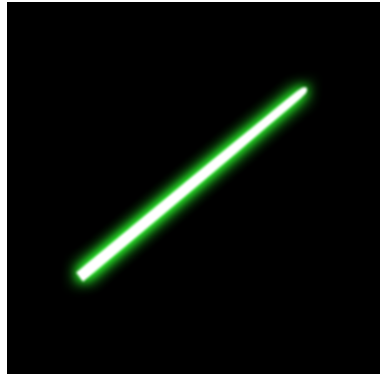


Lightsaber Node v1.0

by Martin Lubich (loramel)



The light_saber node aims to replicate the lightsaber effect of the Star Wars movies. The effect you get by using this node is shown in the picture above.

The design of this node is such, that it should be easy and fast to apply, and should work in a wide range of applications.

The node offers the following features

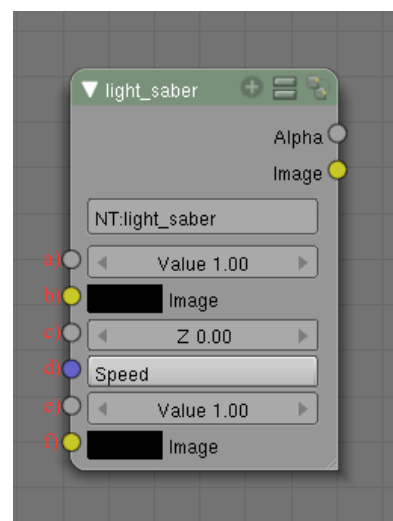
- easy definition of the sabers color by feeding a RGB value to the color socket
- possibility to use motion blur to get wide areas when the saber is moving fast
- adjustable amount of the motion blur effect
- adjustment to accommodate to the relative size of the saber visible in the image
- works relatively independent of image resolution

This document describes the usage and the inner workings of the light_saber node, so that you are able to modify it to your own desire.

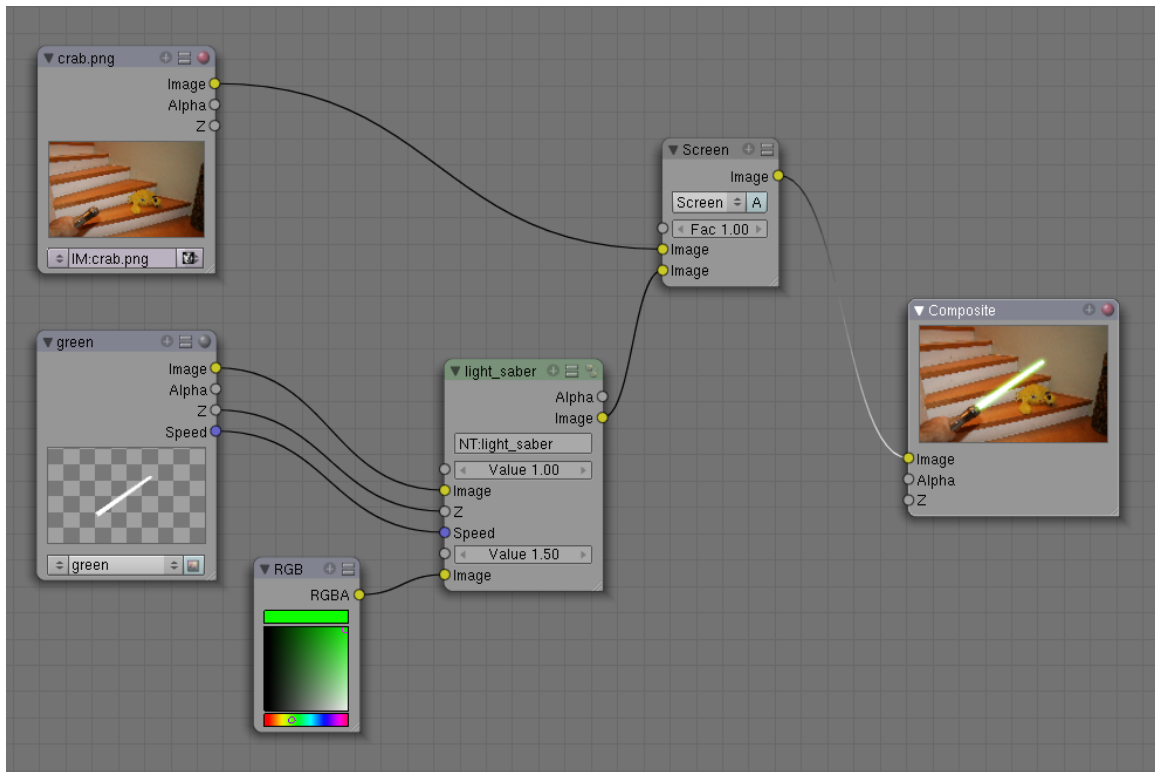
You can append the node to your own blend file from the supplied blend file light_saber.blend. You find it under NodeTree/light_saber. Once appended you can add it to your node editor through Add/Group/light_saber.

The several inputs control the following aspects

- a) controls the motion blur amount (0.0 – 1.0)
- b) image of your saber mask
- c) Z value of your saber mask
- d) speed of your saber mask
- e) controls the glow amount relatively to the image size
- f) color of the saber glow



A typical application is setup the following way



You typically have a render layer with the mask of your light saber. (The supplied blend file has an importable saber mask with animation controls for easy rotoscoping). The renderlayer must have the vector blur attribute set, since the speed value is needed to be fed to the light_saber node, even if you do only stills. If you omit the speed socket you wont see the saber.

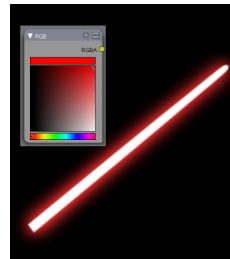
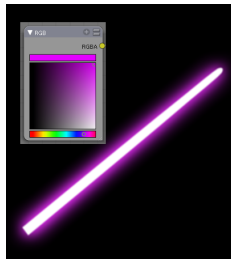
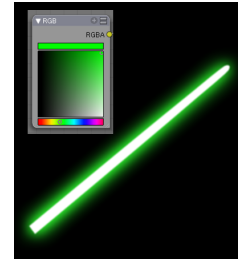
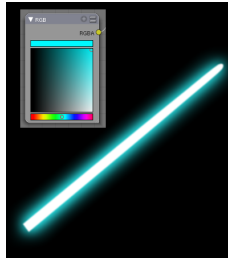
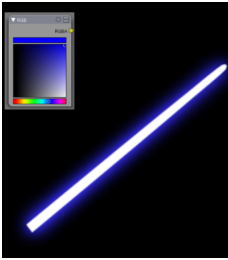
You add a RGB node and set it to the desired color. It is advisable to use fully saturated colors for best effect, but of course you are free to experiment with any color you want.

And finally you will have the actual image where the lightsaber effect will be composited into. The mixing is done with a color node in Screen mode with a factor of 1.0. Higher values tend to make the saber effect brighter but also less convincing.

Usage - Description of the input sockets

f) Color Input

Below you see a collection of typical lightsaber colors. The only variable was the input color. No other parameters had to be adjusted.



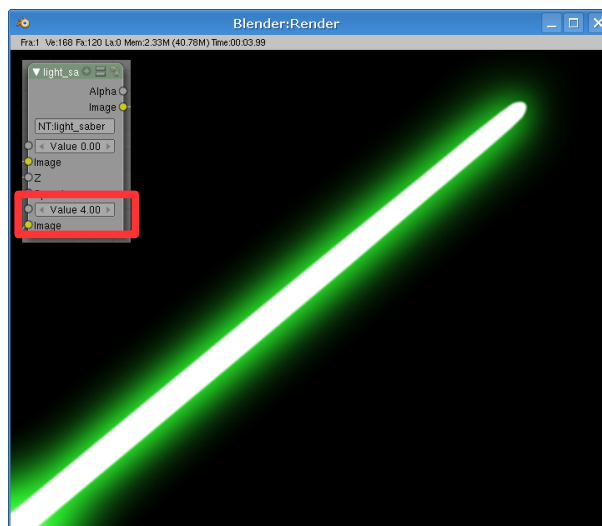
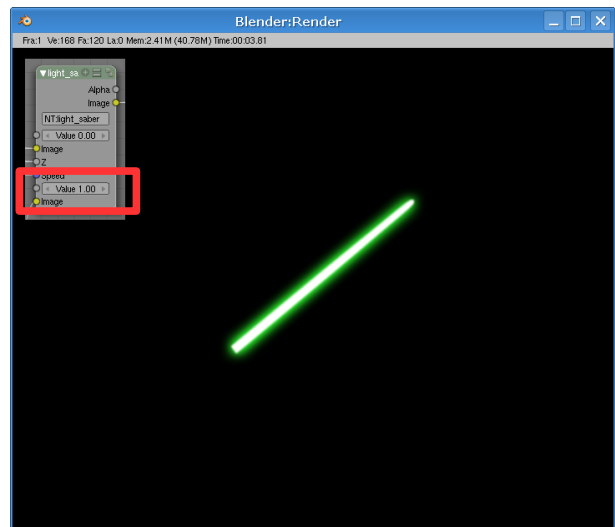
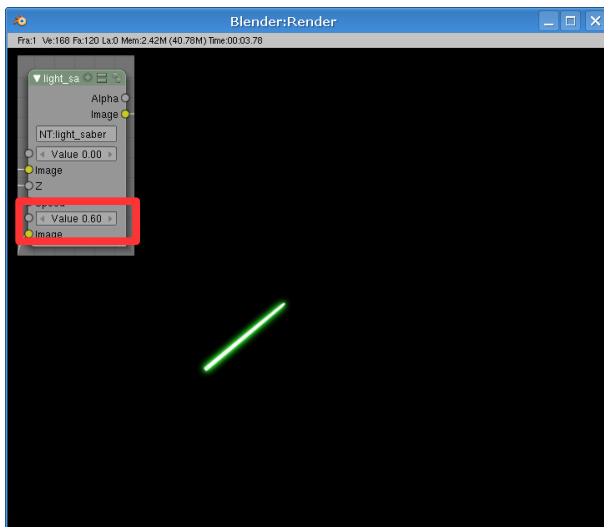
b), c) and d) Saber Mask input

You have to supply the shape of the saber as pure white along with the Z and the Speed Values. The Z and Speed Sockets are used to be able to generate the vector blur effect. Even if you do not need this effect you have to connect to these sockets, otherwise you will not see any lightsaber effect.

e) *Glow amount relativ to the image size*

Depending on the relativ size of the saber in your final image you have to use this parameter to tune the effect to be plausible. The glow around the saber has to be much broader if the saber itself is big compared to the image area. The node can account for different absolute image sizes to control the actual blurring, which underlies the effect, but until now I have no solution for extracting the relativ size information for the saber.

Below you see 3 different relativ saber sizes and the according size settings for the node:



The strong glow effect always has approx. the same size as the white saber blade itself. Adjust this value until you get a correct glow for the size of your saber. Since this is a normal input parameter, it can easily animated through the use of the time node, thus seamlessly adjusting the effect whenever the saber move nearer or farther away.

a) Motion Blur control

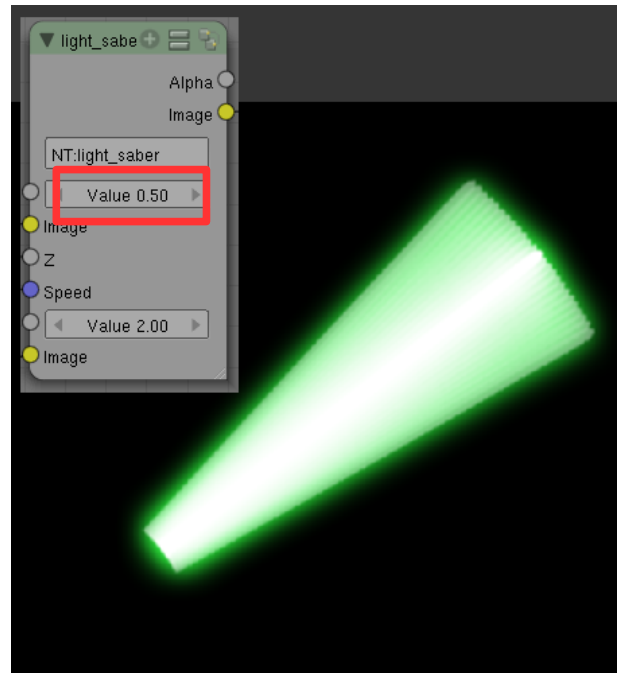
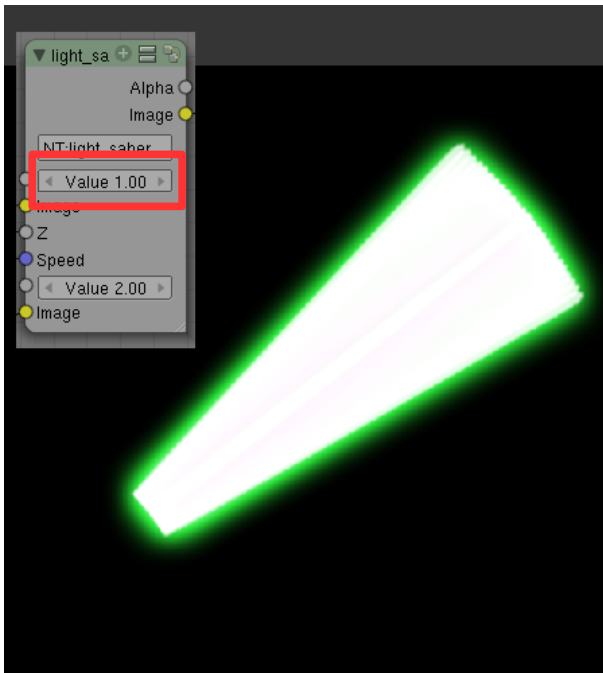
When animating/rotoscoping lightsabers you have two options for faking the motion blur effect:

- Adjust and animate the shape of the saber mask to make it appear wider.
- Use blenders vector blur filter to create the desired effect

Using the first method you have more control over the shape of your blur, but will always have a solid white area for the saber.

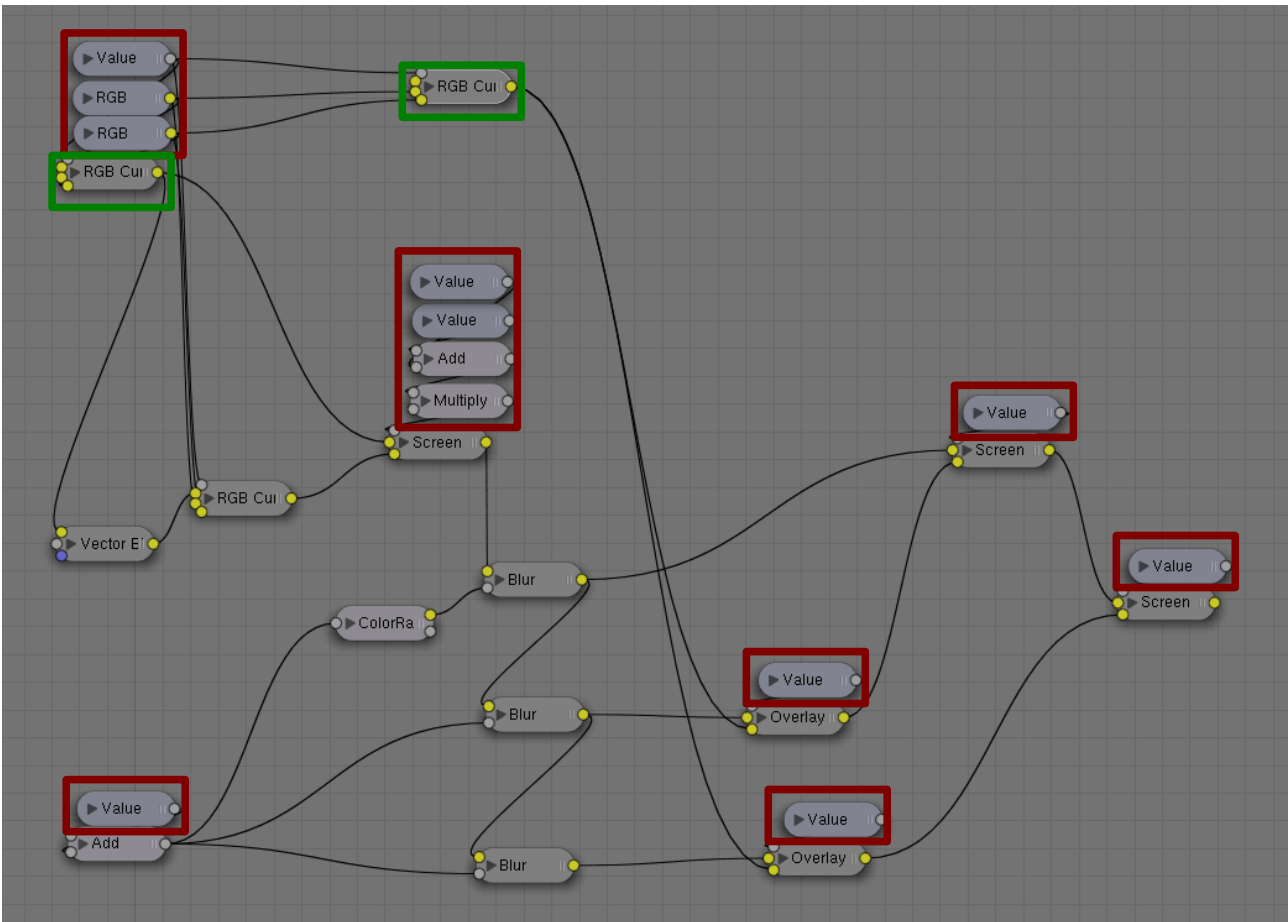
The light_saber node supports the second variant but lets you control how strong the vector blur effect is applied.

See below for an example of the settings 1.0 and 0.5. You should not use values below 0.0 or above 1.0, as you will get strange effects.



Node Setup – Inner Workings

Upon entering the light_saber node you get the following setup



This looks more complex than it is. The node with the dark red boxes around are just providing constant values as input parameters to the actual working nodes like blur, screen overlay etc. I have done this, that these parameters do not show up as input sockets in the lightsaber node, so the interface keeps clean and understandable. Also the RGB Curve nodes with inside the green rectangles function only as provider for an input socket (image, and color) as well as a multiplier for the same signal to different internal sockets. They do not change the fed images.

On the next page you see an expanded and stripped version of the light_saber node.

The blue rectangle show the nodes for the lightsaber effect.

Lets follow the flow of the inputs to see how this works.

- The lightsaber mask image along with its Z and speed values are fed into a vector blur node, which is set to curved and has an effect strength of 1.0. This will produce the raw burred image of the saber. If the speed vector is 0, no blurring will occur, and the saber image will be left unchanged.
- The vector blurred image is fed into a RGB Curves node to increase the contrast, so that the whole blurred area is a mostly solid white.
- The result of the enhanced vector blurred image is than screened with the original saber image. The amount of the screen operation is controlled from the outside. This is the motion blur control (a)
- The result from the screening is than fed into a chain of 3 blur nodes, each with increasing blur radii and each getting as input the result of the former blur operation. The blur type is set to Mitch to get a more defined shape. This accomplishes 3 layers of an increasing (in size) and decreasing (in luminance) glow, but still without color.
- Color is introduced only to the last two blur stages, by overlaying the supplied color (parameter f) to the blurred stage. The amount of the overlay operation is fixed inside the light_saber node. With this you can adjust the saturation and balance of the outer and inner glow.
- The first blur stage does not get any color. This stage creates the slight blurry edge of the completely white blade.
- As you can see, all the blur amounts are specified as relative amounts in respect to the images X and Y size. The size input of each blur node is connected to an external available parameter (parameter e) and controls the actual blur amount. There is one specialty with this concerning the first blur stage. In order to get not too blurred blade edges, the total amount of blurring should not exceed a certain value, regardless of what is fed from the outside. To achieve this, the external value is fed into a ColorRamp node, with linear setup. This node will effectively cut off all values above 1 and will never give a value above 1, exactly what we need.
- The 3 blur stage are combined again using the screen operation. The mix amount is intenally fixed (to the values you see here) and can be used to control the balance between the individual glows.

These are quite some operations to be done for a single lightsaber effect, but the really time consuming parts are the blurring of the saber image. The blur operation can take quite some time, especially if you are rendering to a high target resolution and using a high relative size factor.